Outline:
JavaScript Overview
- When/where
- Designer
- Purpose
- Derivatives (JScript, etc)
- Standardization (ECMA)
- Typing (duck typing, with some formal analysis)
- Paradigm(s) (functional?, imperative, object-oriented (prototype-based) )
- First-class functions
- Operator precedence
- Major revisions and additions

JavaScript Prototypes
- What's a prototype (dynamic objects, “cloning” explained, direct instance vs. template, similarity to associative arrays)
- Example (simple Person object)
- Compare/contrast to class-based objects in Java (Featherweight Java?)
  - Constructors, attributes, functions, etc.
- Java example of class-based Person object
- Use formal techniques from latest homework to analyze Java/JavaScript classes
- Prototype Inheritance vs. Java Inheritance
  - also analyze formally
  - example programs (instanceOf operator can be used to show inheritance)

Resources:
JavaScript History:

JavaScript is functional, imperative, and object oriented (prototype based):
http://en.wikipedia.org/wiki/Multi-paradigm

JavaScript operator precedence:

Object creation with functions, dynamic objects explained: http://www.xul.fr/javascript/language.html

Similarity to associative arrays / Direct instance vs. template for object:
http://www.w3schools.com/js/js_objects.asp


More from this guy: http://www.crockford.com/javascript/
Example of a person template:

```html
<html>
<body>

<script type="text/javascript">

function person(firstname, lastname, age, eyecolor) {
    this.firstname = firstname
    this.lastname = lastname
    this.age = age
    this.eyecolor = eyecolor

    this.newlastname = function(new_lastname) {
        this.lastname = new_lastname
    }
}

myFather = new person("John", "Doe", 50, "blue")
myMother = new person("Mary", "Doe-Smith", 49, "green")
myMother.newlastname("Smith")

document.write(myFather.firstname + " is " + myFather.age + " years old.\n")
document.write(myMother.lastname + " is " + myMother.age + " years old."

</script>

</body>
</html>
```